

Reading: Basic JavaScript concepts

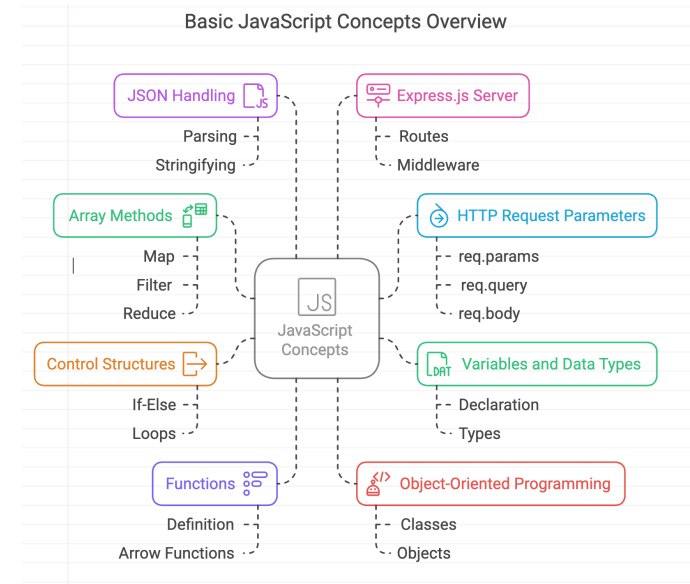
Estimated Time: 30 minutes

Learning objectives

- Describe variable declaration and data types in JavaScript
- Apply control structures for logic flow
- Define and use functions, including arrow functions
- Explain the working of array methods (map, filter, reduce) for data transformation
- Implement basic object-oriented programming with classes
- Describe parse and stringify JSON data
- Demonstrate how to set up a simple Express.js server and define basic routes
- Explain how to use req.params, req.query, and req.body in Express

Prerequisite

As highlighted in the [Course Overview](#), basic knowledge of JavaScript is a prerequisite for this course. This reading includes additional JavaScript concepts to support and enhance your understanding.



Variables and data types

JavaScript uses let, const, and var to declare variables.

```
let name = "Book Review"; // string
const rating = 4.5; // number (immutable binding)
var isPublished = true; // boolean (function-scoped)
let book = { title: "JS Guide", author: "MDN" }; // object
let tags = ["JavaScript", "Express"]; // array
```

Variable Declarations Comparison			
Characteristic	`let`	`const`	`var`
Mutability	Mutable	Immutable	Mutable
Scope	Block	Block	Function
Initialization	Required	Required	Not Required

Control structures

Control structures control logic flow in programs.

```
// Conditional
if (rating > 4) {
  console.log("Highly rated!"); // executes if condition is true
}
```

```
} else {  
  console.log("Needs improvement."); // executes otherwise  
}  
// Loop  
for (let tag of tags) {  
  console.log(tag); // prints each tag  
}
```

Functions

Functions encapsulate code logic for reuse.

```
// Function Declaration  
function greet(user) {  
  return `Hello, ${user}`; // returns a greeting  
}  
// Arrow Function  
const add = (a, b) => a + b; // concise syntax for function  
console.log(greet("Dev")); // Hello, Dev
```

Array methods

Used to process data collections.

```
const books = [  
  { title: "JS Basics", rating: 5 },  
  { title: "Node Intro", rating: 3 }  
];  
// Filter: Get books with rating > 4  
const topBooks = books.filter(book => book.rating > 4);  
// Map: Get an array of book titles  
const titles = books.map(book => book.title);  
// Reduce: Calculate average rating  
const averageRating = books.reduce((sum, book) => sum + book.rating, 0) / books.length;  
console.log(topBooks); // [{ title: "JS Basics", rating: 5 }]  
console.log(titles); // ["JS Basics", "Node Intro"]  
console.log(averageRating); // 4
```

Object-oriented JavaScript

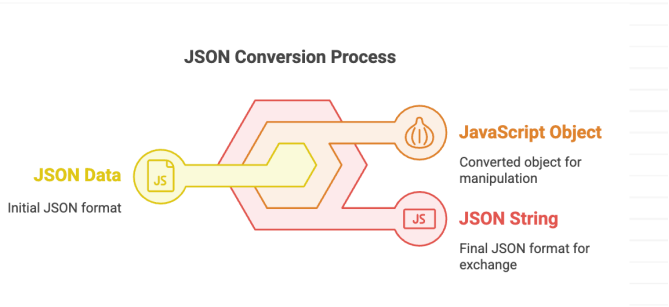
Organizes code using classes and objects.

```
class Book {  
  constructor(title, rating) {  
    this.title = title;  
    this.rating = rating;  
  }  
  describe() {  
    return `${this.title} has a rating of ${this.rating}`;  
  }  
}  
const b1 = new Book("Node Basics", 4.2);  
console.log(b1.describe());
```

Working with JSON

Essential for APIs—data is exchanged in JSON format.

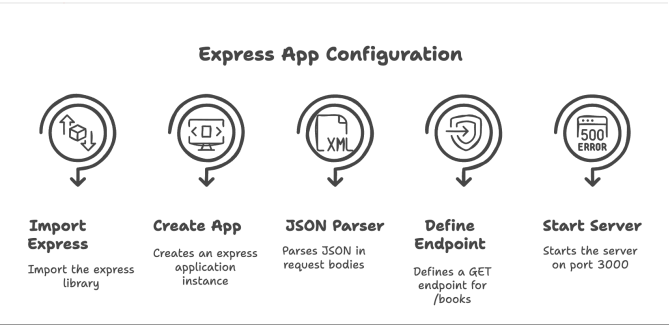
```
let jsonData = '{"title":"Express 101","rating":5}';
const bookObj = JSON.parse(jsonData); // convert JSON to JS object
const newJson = JSON.stringify(bookObj); // convert JS object to JSON
```



Simple Node.js + Express example

Create a minimal server that responds with JSON data.

```
// app.js
const express = require("express");
const app = express();
app.use(express.json()); // to parse JSON bodies
app.get("/books", (req, res) => {
  res.json([ { title: "Learn Node", rating: 4 } ]); // return JSON response
});
app.listen(3000, () => console.log("Server running on port 3000"));
```



HTTP methods overview (REST API)

Used for CRUD operations in web services.

Method	Purpose
GET	Read data
POST	Create data
PUT	Update data
DELETE	Remove data

Understanding req.params, req.query, and req.body

These are ways to access incoming request data in Express.

1. req.params

```
app.get('/users/:id', (req, res) => {  
  const userId = req.params.id; // extract dynamic part from URL  
  res.send(`User ID is ${userId}`);  
});
```

Visiting `/users/123` sets `req.params.id` to `"123"`

2. req.query

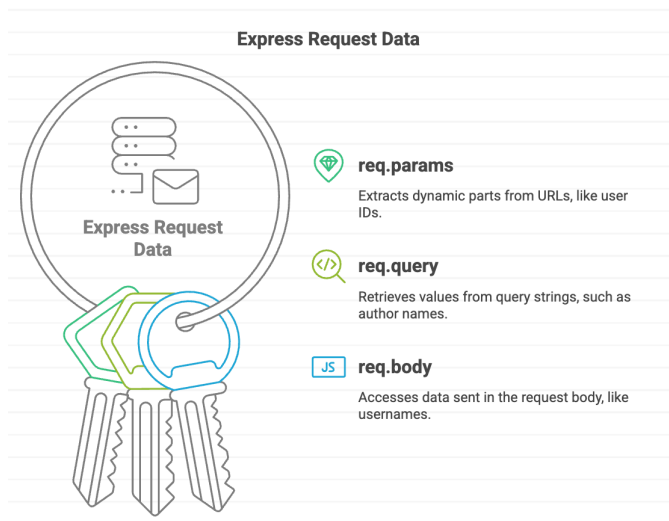
```
app.get('/books', (req, res) => {  
  const author = req.query.author; // extract ?author= value from query string  
  res.send(`Filter by author: ${author}`);  
});
```

Visiting `/books?author=JohnDoe` sets `req.query.author` to `"JohnDoe"`

3. req.body

```
app.post('/register', (req, res) => {  
  const username = req.body.username; // get value from request body  
  res.send(`Username received: ${username}`);  
});
```

For JSON `{ "username": "aname", "password": "pwd123" }`, `req.body.username` returns `"aname"`



Summary

In this reading, you explored the core JavaScript concepts essential for server-side development with Node.js and Express. These included variables, functions, array methods, classes, and working with JSON. You also gained an understanding of how a basic Express server operates and how to handle incoming request data using `req.params`, `req.query`, and `req.body`.

Author(s)

[Ritika Joshi](#)



Skills Network